Hierarchical Clustering: Building a Tree of Similarity

Ahmed BADI
ahmedbadi905@gmail.com
linkedin.com/in/badi-ahmed

December 19, 2025

Abstract

Hierarchical clustering is a family of unsupervised learning methods that builds a hierarchy of nested clusters instead of a single flat partition. Rather than asking "How many clusters exist?" upfront, hierarchical algorithms construct a tree structure (dendrogram) that reveals groupings at multiple levels of granularity. This article provides an accessible yet rigorous introduction to hierarchical clustering. We explain the intuition of bottom-up (agglomerative) and top-down (divisive) strategies, detail common linkage criteria such as single, complete, average, and Ward's method, and show how dendrograms encode the merging process. We also discuss computational aspects, when hierarchical clustering is preferable to methods like k-means, and typical applications in biology, text mining, and exploratory data analysis. Throughout, the focus is on clear, human-friendly explanations supported by equations and illustrative figures.

Keywords: Hierarchical Clustering, Agglomerative, Divisive, Dendrogram, Linkage Criteria, Ward's Method, Unsupervised Learning.

1 Introduction

Most clustering algorithms produce a *flat* partition of the data: each point is assigned to exactly one of K clusters, and the story ends there. However, many real-world datasets have a naturally hierarchical structure [1], [2]. For example:

- In biology, species are grouped into genera, genera into families, families into orders, and so on.
- In document analysis, articles can be grouped by topic, then subtopic, then specific niche.
- In retail, products can be grouped into categories (electronics), subcategories (phones), then brands.

Hierarchical clustering aims to capture this kind of multi-level organization. Instead of producing only one clustering for a fixed number of clusters K, it constructs a tree of clusters, called a *dendrogram*, that shows how individual points merge into clusters and how clusters merge into larger clusters as we move up the tree.

From this dendrogram, we can obtain a flat clustering at any desired level by "cutting" the tree at a chosen height. This makes hierarchical clustering particularly suitable for exploratory data analysis, where we do not know in advance how many clusters to expect.

In this article, we focus on:

- The general principle of hierarchical clustering.

- Agglomerative (bottom-up) vs. divisive (top-down) strategies.
- Linkage criteria that define inter-cluster distances.
- Reading and interpreting dendrograms.
- Advantages, limitations, and practical use cases.

2 Principles of Hierarchical Clustering

2.1 Agglomerative vs. Divisive Strategies

There are two main flavors of hierarchical clustering [2]:

Agglomerative (bottom-up).

- Start with each data point as its own cluster.
- At each step, merge the two *closest* clusters into a new cluster.
- Continue until all points are merged into a single cluster (the root).

This is by far the most widely used approach in practice.

Divisive (top-down).

- Start with all points in one big cluster.
- Recursively split clusters into smaller ones, typically by applying some flat clustering method within each cluster.
- Continue until each point stands alone or until a desired level of granularity.

Divisive methods can be conceptually appealing but are often more computationally demanding, so they are less common in standard libraries.

2.2 Dendrogram: The Tree of Merges

The output of hierarchical clustering is usually visualized as a **dendrogram**. Conceptually, a dendrogram is a binary tree:

- Leaves at the bottom correspond to individual data points.
- Internal nodes represent merges of two clusters.
- The vertical axis (height) indicates the distance or dissimilarity at which two clusters were merged.

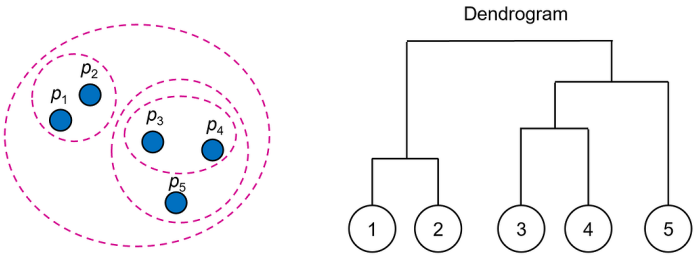

Figure 1: Example dendrogram: each merge is shown at the height corresponding to the inter-cluster distance at merging time. Cutting the tree at a horizontal line yields a flat clustering.

To obtain a flat clustering with K clusters, we can draw a horizontal line across the dendrogram and take the connected components below that line as clusters. Cutting higher in the tree yields fewer, larger clusters; cutting lower yields more, smaller clusters.

3 Agglomerative Hierarchical Clustering

3.1 Basic Algorithm

Given a dataset $X = \{x_1, \ldots, x_n\}$ and a distance measure $d(\cdot, \cdot)$ between points, the agglomerative clustering algorithm proceeds as follows [1]:

Algorithm 1 Agglomerative Hierarchical Clustering

1: Compute the pairwise distance matrix D between all points.
2: Initialize n clusters: each point x_i is its own cluster $C_i = \{x_i\}$.
3: **repeat**
4: Find the two *closest* clusters C_a and C_b according to a chosen **linkage** criterion (Section 3.2).
5: Merge them into a new cluster $C_{ab} = C_a \cup C_b$.
6: Update the cluster distance matrix to reflect distances between C_{ab} and all other clusters.
7: **until** only one cluster remains

The key ingredient in step 4 is the definition of distance between clusters, which is controlled by the linkage criterion.

3.2 Linkage Criteria

Suppose we currently have two clusters A and B, each containing multiple points. We already know how to compute distances between individual points, but how should we define the distance between A and B?

Several standard linkages are used in practice [1], [2]:

Single Linkage.

$$d_{\text{single}}(A, B) = \min_{x \in A,\, y \in B} d(x, y).$$

The distance between two clusters is the smallest distance between any pair of points across clusters. This tends to form "chains" of points and can produce long, stringy clusters.

Complete Linkage.

$$d_{\text{complete}}(A, B) = \max_{x \in A,\, y \in B} d(x, y).$$

Here we look at the two points (one in each cluster) that are farthest apart. This tends to produce compact, spherical clusters and is more robust to chaining, but can be sensitive to outliers.

Average Linkage.

$$d_{\text{average}}(A, B) = \frac{1}{|A||B|} \sum_{x \in A} \sum_{y \in B} d(x, y).$$

This uses the average distance between all pairs of points across clusters and often provides a compromise between single and complete linkage.

Ward's Method. Ward's method defines the inter-cluster distance based on the increase in total within-cluster variance that would result from merging two clusters. For Euclidean data, this can be expressed in terms of cluster sizes and centroids [2]. Intuitively, it merges the two clusters that lead to the smallest increase in the sum of squared errors:

$$\Delta(A, B) = \sum_{x \in A \cup B} \|x - \mu_{A \cup B}\|^2 - \sum_{x \in A} \|x - \mu_A\|^2 - \sum_{x \in B} \|x - \mu_B\|^2,$$

where μ_A, μ_B, and $\mu_{A \cup B}$ are centroids of the respective clusters.

Ward's method often produces clusters similar to those found by k-means, but organized hierarchically.

3.3 A Small Worked Example

Consider four 1D points: $x_1 = 1$, $x_2 = 2$, $x_3 = 8$, $x_4 = 9$, with Euclidean distance.

1. Initial clusters: $\{1\}$, $\{2\}$, $\{8\}$, $\{9\}$.
2. Distances:
$$d(1,2) = 1,\ d(1,8) = 7,\ d(1,9) = 8,\ d(2,8) = 6,\ d(2,9) = 7,\ d(8,9) = 1.$$
3. Single linkage: closest pairs are $(1, 2)$ and $(8, 9)$ with distance 1. Suppose we first merge $(1, 2)$ into C_{12}.
4. New clusters: $C_{12} = \{1, 2\},\ \{8\},\ \{9\}$.
5. Distances (single linkage):
$$d(C_{12}, 8) = \min\{d(1,8), d(2,8)\} = 6, \quad d(C_{12}, 9) = \min\{8, 7\} = 7, \quad d(8,9) = 1.$$
6. Next merge: $(8, 9) \to C_{89}$.
7. Finally, merge C_{12} and C_{89}.

This sequence of merges can be drawn as a simple dendrogram, where the lower merges (within each pair) occur at height 1, and the final merge at height 6.

4 Divisive Hierarchical Clustering

In contrast to the bottom-up strategy, **divisive** clustering starts with all points in a single cluster and recursively splits clusters into smaller ones [2]. A general outline is:

1. Start with one cluster containing all data.
2. Choose a cluster to split (often the one with largest diameter or variance).
3. Apply a flat clustering method (e.g. k-means with $K = 2$) to split it into two subclusters.
4. Repeat steps 2–3 until each cluster has only one point or another stopping criterion is met (e.g. maximum depth, minimum cluster size).

Divisive methods can, in principle, produce different hierarchies than agglomerative methods, especially when using sophisticated splitting criteria. However, they are generally more computationally expensive, since each split requires solving a clustering problem on a subset of the data.

In practice, most standard implementations (such as those in common Python or R libraries) focus on agglomerative clustering due to its simpler and more direct formulation.

5 Advantages, Limitations, and Use Cases

5.1 Advantages of Hierarchical Clustering

- **No need to pre-specify K.** The dendrogram provides a full hierarchy of clusters. We can choose the number of clusters later by cutting the tree at a chosen level.
- **Multi-scale structure.** Hierarchical clustering reveals groupings at different levels of granularity, which can be crucial for understanding complex data (e.g. biology, taxonomy, topic hierarchies).
- **Flexible linkage choices.** By selecting different linkage criteria, we can adapt the clustering behavior to the shape and compactness of clusters we expect.

5.2 Limitations

- **Computational cost.** Standard agglomerative algorithms have $O(n^2)$ time and memory complexity, which becomes problematic for very large datasets.
- **Irreversibility.** Once two clusters are merged, they are never split again (in agglomerative methods). Early mistakes due to noise or outliers can therefore affect the entire tree.
- **Sensitivity to distance and linkage.** Different choices of distance metric and linkage criterion can lead to very different dendrograms, so results must be interpreted carefully.

5.3 Typical Applications

- **Bioinformatics:** clustering genes or samples based on expression profiles to discover groups with similar behavior.
- **Document and topic analysis:** building topic hierarchies and exploring the structure of corpora.
- **Customer segmentation:** when a multi-level segmentation (e.g. coarse segments and finer subsegments) is desired.
- **Exploratory data analysis:** as a first step to visualize how data points relate to each other at different scales.

In many workflows, hierarchical clustering complements methods like k-means: we might first use hierarchical clustering to explore how many clusters seem reasonable, then run a centroid-based method with a chosen K for large-scale deployment.

6 Conclusion

Hierarchical clustering provides a rich and flexible view of data structure by building a full tree of nested clusters instead of committing to a single flat partition. Agglomerative methods start from individual points and repeatedly merge the closest clusters, while divisive methods start from a single cluster and repeatedly split it. Linkage criteria such as single, complete, average, and Ward's method control how inter-cluster distances are measured and therefore shape the dendrogram [1], [2].

Although hierarchical clustering can be computationally demanding for very large datasets, its ability to represent and visualize multi-scale structure makes it an invaluable tool for exploratory analysis in fields ranging from biology to text mining. When used carefully—with appropriate distance metrics, linkage choices, and awareness of its limitations—hierarchical clustering can reveal patterns and relationships that flat clustering methods may obscure.

References

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.